



Tech Note #5: DeviceNet with the Momentum Motion Modules and Allen Bradley PLCs

Application Note By Tim McIntosh
October 26, 2004

Abstract:

In this Tech Note the use of DeviceNet with the I²T Momentum motion controller and an Allen Bradley PLC will be discussed. Some general information will be provided on the DeviceNet communication protocol with a more complete discussion on integrating the I²T Momentum motion module into an Allen Bradley PLC application, and what functions are available on the I²T Momentum motion module over DeviceNet. The I²T Motion Module was designed from the ground up to be controlled over a network, so all functions, commands, and status information is available to the PLC via the network through various user commands.

DeviceNet Communication adapter for I²T Momentum motion module:

The DeviceNet communication adapter from Schneider Electric can be connected to any TSX Momentum base. It provides a direct connection to the DeviceNet network, enabling a programmable controller or other DeviceNet master device to communicate to the Motion module allowing for a PLC or automation application with integrated motion control.

Figure 1 shows the layout of a typical adapter and a Momentum module.

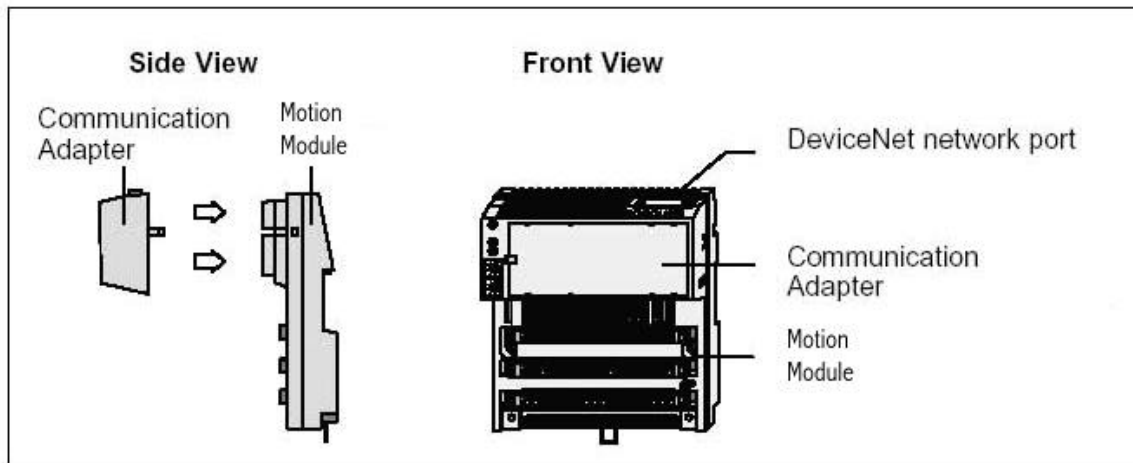


Figure 1 Communication adapter with Momentum Motion Module

221 Seventh Street, Suite LL, Pittsburgh, PA 15238

PH: 412-828-1200 FX: 412-828-0320

WEB: www.isquaredt.com

For further data on the communication adapter specifications, and network compatibility, please refer to Modicon's user guide for this adapter.

The DeviceNet communication adapter has three front panel indicators showing its operating status, and a pair of rotary switches for setting the adapter's network address.

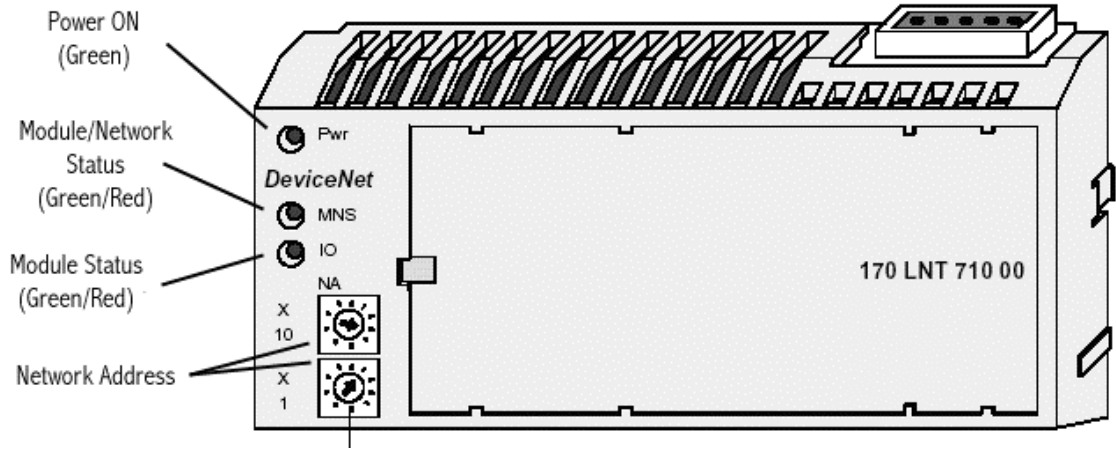


Figure 2 Status Indicators and Network Address

Power ON Indicator:

Indicator State	Status
Green (Steady)	Normal Operation: Power is present from base
Off	Adapter is not receiving power from base

Module/Network Status Indicator:

Indicator State	Status
Green (Steady)	Normal Operation: Adapter communicating on Network
Off	Adapter is not receiving power from base
Green (Flashing)	Adapter does not have an established network connection
Red (Steady)	Unrecoverable fault in network connection
Red (Flashing)	Adapter connection to base has timed out
Red/Green (Flashing)	Network access error

Module Status Indicator:

Indicator State	Status
Green (Steady)	Normal Operation: No Base Faults
Off	Base inactive

Network Node Address:

Do not install any adapter unless you have set its network address for your application

Node Address	Upper Switch	Lower Switch
0 .. 9	0	0 .. 9
10 .. 19	1	0 .. 9
20 .. 29	2	0 .. 9
30 .. 39	3	0 .. 9
40 .. 49	4	0 .. 9
50 .. 59	5	0 .. 9
60 .. 63	6	0 .. 3

Connecting the adapter to the Network:

The adapter has a 5-pin male connector for connection to the network. Its pin-to-pin spacing is 5.00 mm. A suggested mating connector is: Schneider Automation part 170 XTS 060 00.

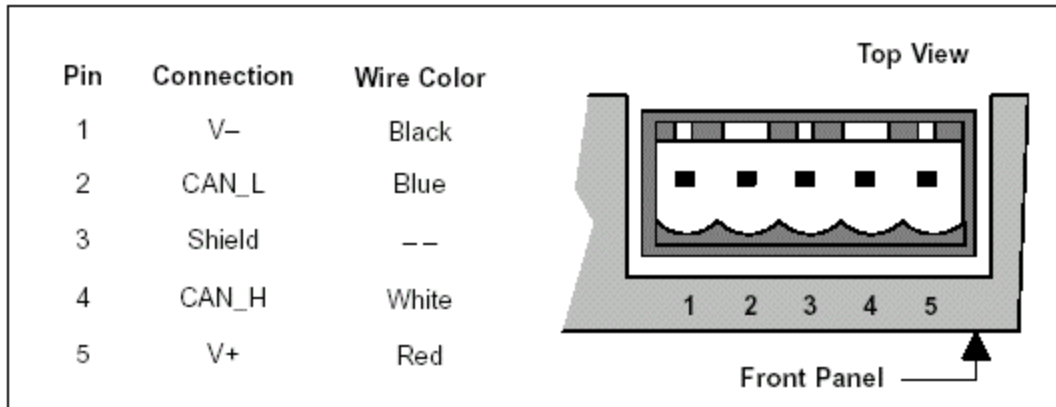


Figure 3 Network Connection

DeviceNet requires a termination resistor for the last node on a network cable. The adapter has a jumper to provide this terminating impedance to the network cable.

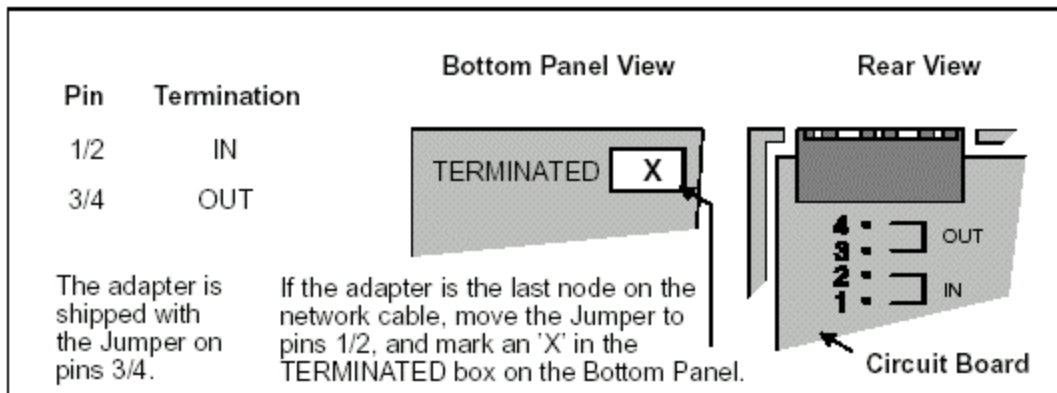


Figure 4 Network Terminator Jumper

Allen Bradley PLC setup for communicating to the I²T Motion Module

Scanner Setup:

The PLC will need a scanner module like the 1747-SDN scanner module. This module provides 32 input and output words for network communication over DeviceNet. Specific instructions to set this card up on the PLC should be referenced from the PLC user manual. Setup requirements specific to the I²T Motion Module will be provided here.

I/O Setup:

The I²T Motion Module uses 8 binary (BIN) bi-directional registers. On the Allen-Bradley PLC 8, consecutive Input registers, and 8 consecutive output registers will need to be allocated from the available registers assigned to the scanner module. It is recommended that the scanner be set to poll this node on some predefined time, such as every 10 msec, as this will simplify the PLC application.

For example, a scanner module may have inputs words I:1.0 through I:1.31, and output words O:1.0 through O:1.31 mapped to it for communicating over the DeviceNet Network. So, choosing 8 consecutive registers for communication may have the following result:

I/O Map Registers

I/O	Description	I/O	Description
I:1.0	Reserved	O:1.0	Reserved
I:1.1	Control Register	O:1.1	Status Register
I:1.2	Data to I ² T Motion Module	O:1.2	Data from I ² T Motion Module
I:1.3	Data to I ² T Motion Module	O:1.3	Data from I ² T Motion Module
I:1.4	Data to I ² T Motion Module	O:1.4	Data from I ² T Motion Module
I:1.5	Data to I ² T Motion Module	O:1.5	Data from I ² T Motion Module
I:1.6	Data to I ² T Motion Module	O:1.6	Data from I ² T Motion Module
I:1.7	Command Register	O:1.7	Echo Register

Integer Register Setup:

Once the scanner has mapped the I/O registers to the I²T Motion Module the bits should be copied, or moved to integer registers in the PLC allowing the PLC application to send commands to the module by setting integer values. The 8 input words should be copied to 8 consecutive integer registers using either the PLC move block “MOV” or copy file block “COP”. However, the order of the words should be reversed, as this will allow a more clear reference to the I²T Motion Module documentation for user commands, and status feedback.

So, if the input words are to be copied to Integer registers N20:1 through N20:8 then the following inversion should occur:

Invert memory locations:

Input Words		Output Words	
Date In	Moved To	Data In	Moved To
I:1.7	N20:1	N20:9	O:1.7
I:1.6	N20:2	N20:11	O:1.6
I:1.5	N20:3	N20:12	O:1.5
I:1.4	N20:4	N20:13	O:1.4
I:1.3	N20:5	N20:14	O:1.3
I:1.2	N20:6	N20:15	O:1.2
I:1.1	N20:7	N20:16	O:1.1
I:1.0	N20:8	N20:17	O:1.0

Integer Mapped I/O Description:

The following describes the Integer register mapping to the modules based on the previous movement of the I/O words to these integer registers. The first 6 input and 6 output registers are used to send and receive data back-and-forth to the I²T Motion Module. The 7th input and output registers are used to send control and status information to the module. And, the 8th input and output registers are used to send I/O control and status to the module.

Integer Registers

Register	Description	Register	Description
N20:1	Command Register	N20:9	Echo Register
N20:2	Data to I ² T Module	N20:10	Data from I ² T Module
N20:3	Data to I ² T Module	N20:11	Data from I ² T Module
N20:4	Data to I ² T Module	N20:12	Data from I ² T Module
N20:5	Data to I ² T Module	N20:13	Data from I ² T Module
N20:6	Data to I ² T Module	N20:14	Data from I ² T Module
N20:7	Control Register	N20:15	Status Register
N20:8	User Outputs	N20:16	Axis and User Inputs

Control and Status Register Bits

Control Register		Status Register	
Bit	Description	Bit	Description
Bit 1	Clear Faults	Bit 1	Axis 1 Faulted
Bit 2	N/A - Axis 1 Enable	Bit 2	N/A - Axis 1 Enabled
Bit 3	N/A - Axis 1 Disable	Bit 3	N/A - Axis 1 Homed
Bit 4	N/A - Axis 1 Home	Bit 4	N/A - Axis 1 Jogging+
Bit 5	N/A - Axis 1 Jog+	Bit 5	N/A - Axis 1 Jogging-
Bit 6	N/A - Axis 1 Jog-	Bit 6	Reserved
Bit 7	Axis 1 Zero Position	Bit 7	Reserved
Bit 8	Reserved	Bit 8	Reserved
Bit 9	Reserved	Bit 9	Axis 2 Faulted
Bit 10	N/A - Axis 2 Enable	Bit 10	N/A - Axis 2 Enabled
Bit 11	N/A - Axis 2 Disable	Bit 11	N/A - Axis 2 Homed
Bit 12	N/A - Axis 2 Home	Bit 12	N/A - Axis 2 Jogging+
Bit 13	N/A - Axis 2 Jog+	Bit 13	N/A - Axis 2 Jogging-
Bit 14	N/A - Axis 2 Jog-	Bit 14	Reserved
Bit 15	Axis 2 Zero Position	Bit 15	Reserved
Bit 16	Reserved	Bit 16	Reserved

User Output & Input Register Bits

User Output Registers		User Input Registers	
Bit	Description	Bit	Description
Bit 1	Reserved	Bit 1	Reserved
Bit 2	Reserved	Bit 2	Reserved
Bit 3	Reserved	Bit 3	Reserved
Bit 4	Reserved	Bit 4	User Input 5
Bit 5	Reserved	Bit 5	User Input 4
Bit 6	Reserved	Bit 6	User Input 3
Bit 7	Reserved	Bit 7	User Input 2
Bit 8	Reserved	Bit 8	User Input 1
Bit 9	User Source 2	Bit 9	Axis 2 Fault Input
Bit 10	User Source 1	Bit 10	Axis 2 Home Input
Bit 11	Allow PLC to set User Outputs	Bit 11	Axis 2 Rev Input
Bit 12	User Output 5	Bit 12	Axis 2 Fwd Input
Bit 13	User Output 4	Bit 13	Axis 1Fault Input
Bit 14	User Output 3	Bit 14	Axis 1Home Input
Bit 15	User Output 2	Bit 15	Axis 1Rev Input
Bit 16	User Output 1	Bit 16	Axis 1Fwd Input

Module User Commands:

The I²T Motion Modules were designed from the ground up to be integrated into a networked control system. There are currently over 300 user commands for module setup, control, and status, and axis setup, control, and status. The I²T motion module's base version will provide all of the functions necessary for the low-level control of 2 axes. With the expansion of the motion module to include the MPL upgrade, the module can contain some user-defined logic at its application level. So, the PLC can have application level control of the module's axis at the PLC level, or with the MPL upgrade, the motion module can run its own user defined application with the PLC acting as a supervisor controller. This distributed architecture allows the PLC applications complexity and size to be reduced, by offloading functions that are difficult to perform in ladder logic to the structured text programming environment of the MPL, while maintaining tight control of the process via the PLC due to the integrated nature in which the motion modules communication protocols were developed.

For example in the MPL application, complex math problems, not related to axis movement, can be performed with results sent back to the PLC, reducing the PLC code complexity, or sequences of axis movements can be programmed in the MPL, with the PLC settings coils to indicate the desired move type.

A complete list of these 300 commands can be found by referencing the help files included in the I²T HMI Module Interface software. This software can be downloaded for free from www.isqauredt.com. The user commands allow the PLC to set everything from low-level control such as PID loop variables or User Units, to high-level supervisory control such as inhibiting program execution, or commanding motion.

Some example commands, and how they might be implemented on the PLC will be provided here to illustrate how the I²T motion module can integrate motion into your PLC application.

A typical command the PLC might set is the Axis JOG speed. This is performed by setting the integer registers that are copied to the I/O registers mapped to the motion module using the following format.

Set Jog Speeds

Write Registers		Response Registers	
Register	Description	Register	Description
NX:X+0	2044 Command	NX:X+0	2044 Command
NX:X+1	Axis 1 or 2	NX:X+1	Axis 1 or 2
NX:X+2	Jog Speed High Word	NX:X+2	Jog Speed High Word
NX:X+3	Jog Speed Low Word	NX:X+3	Jog Speed Low Word
NX:X+4	N/A	NX:X+4	N/A
NX:X+5	N/A	NX:X+5	N/A

So, for example, if the PLC application, using the integer registers defined in the previous section, needs to set the JOG speed for Axis 1 to 10.55 inches per second, then it would set these integer registers to the following values:

Set Jog Speeds Write Registers

Write Registers	
Register	Description
N20:1	2044
N20:2	1
N20:3	10
N20:4	55
N20:5	N/A
N20:6	N/A

Then to confirm the module has received the command, the response registers can be checked until they echo the write registers, i.e.

Check Jog Speeds Response Registers

Response Registers	
Register	Description
N20:8	2044
N20:9	1
N20:10	10
N20:11	55
N20:12	N/A
N20:13	N/A

So, by comparing the response register to the write register, the PLC will have an indication as to when another user command can be sent, and a confirmation that the command was both received, and accepted, by the motion module. In the event that a user command is sent with incorrect data, the response registers will not echo the user command, allowing a break in the PLC application to reduce debug during initial machine programming.

Another user command definition is the read axis positions command.

Read Axis Positions

Write Registers		Response Registers	
Register	Description	Register	Description
NX:X+0	1020 Command	NX:X+8	1020 Command
NX:X+1	N/A	NX:X+9	Axis 1 Position High Word
NX:X+2	N/A	NX:X+10	Axis 1 Position Low Word
NX:X+3	N/A	NX:X+11	Axis 2 Position High Word
NX:X+4	N/A	NX:X+12	Axis 2 Position Low Word
NX:X+5	N/A	NX:X+13	N/A

Note that in this command information is retrieved from the module, rather than being sent to the module. The first number of the command word annotates the direction of information flow. All 1XXX commands request data from the module, while all 2XXX commands send data to the module.

For Example:

A 1020 command can be placed in register NX:X+0 and the positions of Axis 1 and Axis 2 can be read back into register NX:X+9 thru NX:X+12. The enable and disable bits (NX:X+6, Bits 2, 3, 10, 11) should be pulsed until the enabled bits (NX:X+14, Bits 2, 10) turn on or off. The axis home bits (NX:X+6, Bits 4, 12) can also be pulsed. The axis homed bits (NX:X+14, Bits 3, 11) will turn on after completion of the homing sequence. The axis homed bits will reset on power up or when the axis home bits are set. The homing sequence is setup on the HMI Windows Setup Software **Axis** dialog box, or via user commands from the PLC. When the jog bits (NX:X+6, Bits 5, 6, 13, 14) are on, the axis will jog in the direction indicated by the bit. **Note:** The jog+ and jog- cannot be on at the same time.

MPL Integration:

For applications running an MPL application, the PLC has supervisor control of the module, allowing for an integrated, distributed control architecture. The PLC can start and stop any of the numbered application programs in the Motion Module via the user commands, as well as transfer data back-and-forth to the module via registers.

A complete description of the MPL language, and the user commands related to controlling the application programs in the module can be obtained from the www.isquaredt.com.